UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/924,272	08/07/2001	Paul Metzgen	174/193	4896

36981 7590 08/13/2004

FISH & NEAVE
1251 AVENUE OF THE AMERICAS
50TH FLOOR
NEW YORK, NY 10020-1105

EXAMINER	
VU, TUAN A	

ART UNIT	PAPER NUMBER
2124	

DATE MAILED: 08/13/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/924,272	METZGEN, PAUL	
	Examiner	Art Unit	
	Tuan A Vu	2124	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *07 August 2001*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-37* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-37* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *10/5/2001* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date *2003/04/22*.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. This action is responsive to the application filed August 7, 2001.

Claims 1-37 have been submitted for examination.

Claim Rejections - 35 USC § 101

2. 35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

3. Claims 27 and 37 are rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter.

> The Federal Circuit has recently applied the practical application test in determining whether the claimed subject matter is statutory under 35 U.S.C. § 101. The practical application test requires that a " useful, concrete, and tangible result" be accomplished. An "abstract idea" when practically applied is eligible for a patent. As a consequence, an invention, which is eligible for patenting under 35 U.S.C. § 101, is in the "useful arts" when it is a machine, manufacture, process or composition of matter, which produces a concrete, tangible, and useful result. The test for practical application is thus to determine whether the claimed invention produces a "useful, concrete and tangible result".

As per claim 27, the claims recites a method for mapping software constructs into hardware constructs and comprising a set of wires that is mapping a variable. The claim specifies further structural description elaborating on the set of wires, but fails to recite any concrete action as to further specify how the mapping as claimed is being implemented in order to yield an useful result. In addition to only step of mapping a set of wires to a variable, the description as to having one or more wires representing different meaning to the variable does not further limit the mapping step, because this is a structural limitation. Further, the claim fails to provide a concrete and tangible result as required by the practical application test; hence amounts to a mere abstract idea. The claim therefore is rejected for it appears to be directed to a non-statutory subject matter.

As per claim 37, the claim is in connection with claim 27 above, hence is rejected for the same reasons as set forth above.

There are no dependent claims rejected for being dependent on these 2 claims.

Claim Objections

4. Claim 37 is objected to because of the following informalities: The limitation 'a software constructs' and 'the hardware construct' (lines 2-3) need to be corrected, e.g. to become '[a] software constructs' and 'the hardware construct-s-'.

5. Claim 20 is objected because the elements recited as 'the expressions are' (line 6) does not fit to the context of an earlier 'expresssion' recited in line 4; and should be recited as 'the expression is' in order to avoid an indefinite language rejection.

Appropriate correction is required.

Claim Rejections - 35 USC § 102

6. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.
>
> Note: 35 U.S.C. § 102(e), as revised by the AIPA and H.R. 2215, applies to all qualifying references, except when the reference is a U.S. patent resulting directly or indirectly from an international application filed before November 29, 2000. For such patents, the prior art date is determined under 35 U.S.C. § 102(e) as it existed prior to the amendment by the AIPA (pre-AIPA 35 U.S.C. § 102(e)).

7. Claims 1,2, 4-5, 7-8, 24-26, and 34-36 are rejected under 35 U.S.C. 102(e) as being anticipated by Panchui et al., 6226,776 (hereinafter Panchui).

As per claim 1, Panchui discloses a method for generating hardware configuration data from software constructs, the method comprising: parsing high-level software programming code, wherein the code is transparent with regard to the hardware resources and hardware configuration (e.g. Fig. 2; Fig. 3-22 – Note: compiling a C program instructions to provide hardware definition language and register transfer form code is equivalent to parsing and that the C-instructions are abstract data structures remote to the layer of hardware resources, hence transparent to hardware resources and configuration); and compiling hardware configuration data based on the high-level software programming code (e.g. Figs. 3-22).

As per claim 2, Panchui discloses hardware configuration data implementing blocks (e.g. HDL – col. 13, line 1 to col. 14, line 65 – Note: the organizing of hardware circuitry via HDL or Verilog specification implicitly discloses organizing the hardware configuration data in blocks – see Panchui, col. 8, lines 36-40; *always* block – Fig. 8B).

As per claim 4, Panchui discloses pipelining (e.g. Fig. 8B)

As per claim 5, Panchui discloses a FPGA (e.g. col. 12, lines 42-49; col. 13, lines 53-63).

As per claim 7, Panchui disclose runtime decisions for parallel execution (e.g. Fig. 3A-B; Fig. 14A-B – Note: during compiling into hardware configuration, determine which instructions to be configured for being synchronized with parallel processing reads on runtime decisions).

As per claim 8, Panchui discloses C code (e.g. Fig. 2, 3A, 4A, 5A, 6A).

As per claim 24, Panchui discloses mapping software constructs into hardware constructs, comprising mapping into a block of logic operations using a software/hardware compiler (e.g. Fig. 2); coupling input to the block and coupling output to that block (e.g. *Ram32x16, Ram32x1, DivEx* - Fig. 5B1; Fig. 7B3).

As per claim 25 and 26, Panchui discloses wires implementing variables, pointer, expressions, and reset and done signal (Fig. 3-22; *if(reset)* – Fig. 7B3; *state*= ... , *end* – Fig. 7B3) and control flow (*control flow* – col. 7, lines 22-43; Fig. 5A-B).

As per claims 34-36, these are programmable logic resource respective versions of claims 24-26, the programmable limitation being addressed by Panchui's programmable logic (col. 12, lines 42-49; col. 13, lines 53-63); and are rejected with the corresponding rejection in claims 24-26 as set forth therein.

8. Claim 20 is rejected under 35 U.S.C. 102(e) as being anticipated by Killian et al., USPN: 6,477,683.

As per claim 20, Killian discloses optimizing hardware generated by software-to-hardware compiler during compilation, comprising:

locating at least one expression where the expression is used more than once, using a single set of hardware resources to implement such expression (col. 19, lines 47-65; col. 27, lines 10-16 – Note: the fact to replace a instruction of the tree while converting to hardware implementation reads on using a single set of hardware resources to implement the expression); and

selecting at runtime instances that will have access to the single set of hardware resources (Note: the partitioning of hardware in blocks by a HDL specification and compilation inherently

yields runtime instances that will access a set of hardware resources being created from that replacement as taught by Killian).

Claim Rejections - 35 USC § 103

9. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

10. Claims 3, 27, and 37 are rejected under 35 U.S.C. 103(a) as being unpatentable over Panchui et al., 6,226,776, as applied to claim 2 (for claim 3).

As per claim 3, Panchui does not disclose data wires and a computed wire to denote the variable is valid. Panchui discloses wires being mapped to variables and associated pointers to represent a variable, expression or a function (col. 5, line 4-45; Fig. 11A-B) and a set of wires to enable a variable (LCD) with wire specialized to represent data (Fig. 20B2; *input ... A or D or WE, wire [4:0]* - Fig. 5B1); hence has suggested the creating of a wire or pointer structure representing a declared variable or function and wires for data and input control bit. Further, official notice is taken that a set of data being passed through a memory port or a circuit gate and such passage is being enabled by a valid or enable Bit was a known concept in the art of hardware design. Hence, since a set of wires or pointers are used for representing data or variables as suggested by Panchui's HDL and its parsing scheme, it would have been obvious for a skill in the art to represent circuitry wires according to HDL's specification as suggested by Panchui above so that some bit in the set of data (or one wire among the set of wires) would be representing the existence or validity of the variable or a non-null pointer; and that some other

bit/wires represent the data for that variable according to the well-known concept above. The motivation would be because, wires is purported for transporting data through a gate device and this requires control and according to well-known concepts, such control, using one bit or wire to enable or indicate validity, would allow such data wires to be checked by the device or gate before data can be allowed to go beyond such gate, or to be put for storage, or read/written via memory port.

As per claim 27, Panchui discloses a method of mapping software constructs into hardware constructs, comprising mapping a variable into a set of wires (e.g. col. 5, line 4-45; Fig. 11A-B; Fig. 20B2; *input ... A or D or WE, wire [4:0]* - Fig. 5B1). The limitation about the wires representing a variable definition/computation state and the value of the variable in this claim corresponds to that of claim 3; hence is rejected using the same rationale as set forth therein.

As per claim 37, this is the hardware and programmable logic version of claim 27, hence is rejected using the same rationale as set forth therein; the programmable limitation being addressed by Panchui's programmable logic (col. 12, lines 42-49; col. 13, lines 53-63).

11. Claims 6, 9-12, 16-19, and 28-33 are rejected under 35 U.S.C. 103(a) as being unpatentable over Panchui et al., 6,226,776, in view of Killian et al., USPN: 6,477,683 (hereinafter Killian).

As per claim 6, Panchui only discloses optimizing and pipelining of loops (e.g. col. 4, lines 51-63; col. 27, Table, lines 45-55; Fig. 8B); but does not teach hardware capable of speculative execution. The method of applying speculative execution in the art of compiler optimization was a known concept at the time the invention was made. Killian, in a method to compile C-code specifications and convert HDL language into target hardware implementation

analogous to Panchui, discloses pipelining as well as speculative by compiler directives and shared memory (e.g. col. 13, line 44 to col. 14, line 4). It would have been obvious for one of ordinary skill in the art at the time the invention was made to add to the optimization techniques by Panchui the compiler-driven speculation as by Killian because according to Killian, these speculative executions can avert data fetching exceptions which would otherwise require unwanted correction handling resources.

As per claim 9, Panchui discloses a method for generating hardware exploiting parallelism by making decisions at runtime, the method comprising:

generating a control flow in the hardware configured in blocks (e.g. *control flow, HDL, Verilog* – col. 7, lines 22-43; Fig. 5A-B; col. 8, lines 36-40; *always* block – Fig. 8B); making determination to exploit parallel execution based on HDL compilation and configuration, i.e. based on control flow, event or machine state associated with HDL analysis (Fig. 3A-B; Fig. 12A-B; 14A-B – Note: parallel execution and configuration thereof associated with HDL analysis is equivalent to making decision regarding execution of a HDL-derived block from control flow break-down or a machine state and subsequent hardware configuration).

But Panchui does not disclose that the control flow indicates a status for a block, status indicative for a capability for speculation. The concept as to base on an entry tag value, a control bit, a variable value, or predicate check in order to establish whether a chunk of instructions can be executed speculatively at compilation was a known concept in the art of compiler optimization at the time of the invention. The limitation as to provide speculation to the implementation from compiling HDL into hardware target architecture set has been addressed above using Killian; hence this speculation based on a block status while analyzing control flow

as mentioned above by Panchui would have been obvious in light of the known concept and of Killian's teachings for the same reasons as set forth in claim 6 above.

As per claims 10 and 11, the partitioning of HDL constructs into blocks of instructions as evidenced by Panchui (col. 8, lines 36-40; *always* block – Fig. 8B) was a known concept in the HDL programming language (e.g. Verilog, VHDL, Handel-C) and in view of Panchui's teaching of determining to execute simultaneous instructions based on event or machine state analysis or to *always* execute a block (Fig. 3A-B; Fig. 12A-B; 14A-B; Fig. 8A) as put forth in claim 9, the rationale to use an indication at a given point into a block of instructions in order to determine whether to execute the block via speculation has been addressed in claim 9; hence the rationale as to combine Panchui's block with well-known concepts in association with the speculative execution as taught by Killian is herein applied to address why it would be obvious to decide to execute the block based on a tag or predicate statement evaluation.

As per claim 12, the rationale for rejection has been addressed in claim 9.

As per claim 16, Panchui teaches *always* blocks, hence has implicitly disclosed them to be non-mutable operations (e.g. Fig. 12B).

As per claim 17, Killian discloses replacing multiple patterns with a single instructions (col. 19, lines 47-65) and replacing a maximally sized match in the functions tree with one equivalent instruction (col. 27, lines 10-16). Hence, in the line as to enhancing further of Panchui's teaching of a tree traversal and pipeline techniques, it would have been obvious for one of ordinary skill in the art at the time the invention was made to provide on top of the variable mapping and function creation as suggested by Panchui (Fig. 3-22) the replacement techniques (or mutable instruction representing a HDL state being replaced) by Killian because

of the same reasons as to optimize resources so well known at the time the invention was made in the art of compiler where a excess or potential redundant resource usage is averted by a replacement policy or instruction just as taught by Killian.

As per claim 18, Panchui discloses a control flow (e.g. Fig. 5A-B).

As per claim 19, Panchui discloses implementing software in hardware (e.g. *control flow, HDL, Verilog* – col. 7, lines 22-43; Fig. 5A-B; col. 8, lines 36-40; *always* block – Fig. 8B).

As per claim 28, this is a programmable logic resource version of claim 9, the programmable limitation being addressed by Panchui's programmable logic (col. 12, lines 42-49; col. 13, lines 53-63); and is rejected with the corresponding rejection in claim 9 as set forth to address the control flow status for an operation and decision partially based on the control flow limitations.

As per claims 29-33, these claims correspond to claims 10-12, 18, and 19 respectively; hence are rejected using the corresponding rejection as set forth therein respectively.

12. Claims 13-15, and 22-23 are rejected under 35 U.S.C. 103(a) as being unpatentable over Panchui et al., 6,226,776, in view of Killian et al., USPN: 6,477,683; as applied to claims 9, 15, and further in view of Ashar et al., USPN: 6,745,160 (hereinafter Ashar).

As per claims 13-14, Panchui does not explicitly teach deciding to execute the block speculatively and in parallel; or no data dependency between speculative and parallel execution of blocks. But Panchui discloses a tree of function/nodes (Figs. 11) and determining if an external event or a state machine is required during analysis of called function/node derived from HDL and if not then simultaneous execution of C-type functions can be effected (e.g. col. 20, line 22 to col. 21, line 17); and non-dependency of C-type functions being simultaneously

executed (e.g. col. 21, lines 28-46); hence has taught no data dependency of instructions blocks (or functions from a tree node) being paralleled. The motivation as to use speculation in optimization of runtime execution of functions has been addressed using Killian. As for the decision to execute the block in parallel or speculatively, Ashar, in a method using netlist and VHDL to transform loop execution into a optimized and validated sequence or scheduling using speculation and parallelism as mentioned by Panchui and Killian, further discloses pipelining loops and speculative execution being determined upon analysis of control or state graph, loop invariants, and state and boundaries verification (e.g. col. 26, lines 17-29; col. 27, line 45 to col. 28, line16). It would have been obvious for one of ordinary skill in the art at the time the invention was made to add to the combination Panchui/Killian the verification as taught by Ashar so that decision can be made to use either speculation or pipelining or loop unrolling because Ashar's techniques would provide timely validating of data being fetched for execution thus enhance optimizing success without sacrificing resources for data checking and dependencies when it is too late in the execution stage of complex loop execution (see Ashar Background).

As per claim 15, Panchui discloses resource sharing with re-used blocks (col. 5, line 56 to col. 6, line 4) and Killian sharing of a memory portion (col. 13, line 44 to col. 14, line 4). In view of the teachings by Ashar and the non-dependency teaching by Panchui as mentioned above, it would have been obvious for one of ordinary skill in the art at the time the invention was made to provide on top of such non-dependency when running in parallel and/or via speculation the step of sharing a variable in memory or a block as suggested by Panchui and enhanced by Killian to the method of optimization provided by Panchui/Killian and Ashar;

because of the same reasons as to optimize resources so well known at the time the invention was made in the art of loop pipelining and parallel execution of instructions in the art of compiler optimization techniques.

As per claim 21, in reference to claim 15, Panchui discloses using a single set of hardware resources to implement a software program entities represented by a node in a control flow, i.e. a set of hardware resources while converting a HDL function or blocks into hardware implementation (e.g. Fig. 3-22).

As per claims 22 and 23, Panchui/Killian and Ashar implicitly discloses the optimization process, such as pipelining, speculation execution etc. to take place late in the phases of HDL conversion without intervention of the user (re claim 15).

Conclusion

13. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Tuan A Vu whose telephone number is (703)305-7207. The examiner can normally be reached on 8AM-4:30PM/Mon-Fri.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Kakali Chaki can be reached on (703)305-9662.

Any response to this action should be mailed to:

Commissioner of Patents and Trademarks

Washington, D.C. 20231

or faxed to:

(703) 872-9306 (for formal communications intended for entry)

or: (703) 746-8734 (for informal or draft communications, please consult Examiner before using this number)

Hand-delivered responses should be brought to Crystal Park II, 2121 Crystal Drive, Arlington. VA. , 22202. 4th Floor(Receptionist).

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

VAT
August 6, 2004

KAKALI CHAKI
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100